Methodology for independent control of multiple near-surface microscopic agents with uniform magnetic fields

Research Thesis

Presented in Partial Fulfillment of the Requirements for Graduation

With Research Distinction in Physics

In the Undergraduate Colleges of The Ohio State University

Emily Osborne

College of Arts and Sciences

The Ohio State University

May 2021

Project Advisor: Professor Ratnasingham Sooryakumar, Department of Physics

Abstract

Microscale robotics have numerous applications in engineering and the biomedical sciences, as the manipulation of small cells or microparticles is necessary for a variety of tasks. However, scaling robotics down to the microscopic regime is challenging, as traditional methods of applying direct forces to individual agents becomes increasingly difficult. Recent studies have turned to utilizing global, uniform fields to all agents, relying on a heterogenous population of agents to demonstrate differentiated responses. This work defines an algorithm to generate a field sequence capable of simultaneously transporting all agents from specified positions to chosen final positions. Using this algorithm, two demonstrations of quasi-independent control of multiple heterogenous agents at the microscale are presented as a proof-of-concept.

Introduction

Microscale robotics is emerging as an important component in manufacturing [1] and biomedical fields [2]-[4] such as in the assembly of microscopic structures [5], targeted cargo or drug delivery [6]-[9], and in cell diagnostics [10]. However, there are challenges to simultaneously manipulate robotic agents at this length scale due to issues related with scalability and differences amongst agents [11]. The two primary approaches currently used to simultaneous achieve quasi-independent control over multiple micro-scale agents rely on the use of (i) individually directed local forces and (ii) uniform fields applied over macroscopic length scales.

The application of independent localized forces on identical individual agents has been realized by optical trapping [12] or by strategically using nonuniform gradients to control agents

at different locations within the system. These gradients may be chemical [13], electrical [14], ultrasound [15], or magnetic [16]-[19]. Using this latter technique for example, large swarms of magnetic bacteria were directed by local magnetic field gradients to push microscopic blocks and build elementary structures [20]. Such methods however encounter issues with scalability, as it becomes increasingly difficult to provide independent forces targeted on each unit within a large number of small agents. For example, in optical trapping schemes, if each agent is controlled by its own laser, then there is a limit to how many agents can be controlled due to the increasingly complicated laser arrays needed.

The second type of control is via applying globally uniform magnetic or electric fields [21]-[25]. This scheme however is burdened by the inevitable non-identical properties of the agents leading to different responses to the applied fields. While this approach is easier to scale, it is often problematic to engineer synthetic agents to have sufficient heterogeneity so that their individual motions can be appreciably differentiated even under an identical global field. Bacteria offer a potential solution as useful microscopic agents in robotics as they are relatively simple to produce, and often exhibit many properties which have cell-to-cell variability that could be exploited for local guided movement [26]-[31].

In this paper both magnetotactic bacteria (MTB) and superparamagnetic microscopic beads are used as agents in a proof of concept study to achieve controlled nonparallel motion of multiple agents under a global, uniform magnetic field. Both the bacteria and the beads are inherently heterogenous agents. For instance, motile MTB vary in cell shape, velocity, and internal magnetic moment. The synthetic beads also have slight differences in shape and magnetic moment due to the inherent limits of manufacturing precision. For each of these living and synthetic agents, strategically tuning the external field gives rise to unique physics leading to

nonparallel individual responses which depend on the global field, or control signal. By defining a set of “control signals”, or specific magnetic fields, and characterizing the responses of agents to those fields, we developed an algorithm which computes the sequence of control signals that is required to transport multiple agents simultaneously between a chosen initial configuration to a chosen final configuration.

This technique will only be useful if there is sufficient differentiation between agents for a given control signal. This feature is necessary to generate a large set of possible final configurations to enable selecting nearly arbitrarily destinations for each agent’s final position. As long as each agent’s responses are sufficiently nonparallel from others, it is possible to compute a sequence of different magnetic fields which can simultaneously transport all agents to their specified final positions. Our goal is to develop an algorithm which can read in the initial and the chosen final configuration of multiple agents to then output the sequence of field control signals necessary to concurrently carry out this desired multi-agent transport.

While this result can ideally be generalized to any agents exhibiting differentiated motion, the current focus is on using two separate magnetic systems for proof-of-concept tests. The first uses magnetotactic bacteria as the robotic agents, and static, tilted magnetic fields as the control signals, where each signal is defined by the degree of tilt. The second utilizes superparamagnetic beads as agents, with rotating fields instead acting as the control signals. In this case signals are defined using the frequency and axis of rotation and the strength of the magnetic field. In simulations, by determining the sequence of control signals and the duration for which each should be applied, a total of four cells or beads (the agents) are shown to move pseudo-independently from any initial location to their pre-determined final destinations.

Methods and Materials

AMB-1 Bacteria

The first proof-of-concept uses a magnetotactic bacteria (MTB) known as *Magnetospirillum magneticum* strain AMB-1. These bacteria are spirochetes, and thus have a helical shaped cell body. Each has a flagellum at both ends, or poles, of their cell body. One flagellum is active at a time, allowing each bacterium to swim either forward or backward along the line defined by its cell body orientation. These bacteria also synthesize membrane-bound magnetite particles, known as magnetosomes, which form a line between the two poles of its cell body. This gives each bacterium an inherent magnetic moment, and consequently a bacterium will align its cell body, and thus its swimming direction, with any external magnetic field. Therefore cellular motion is directed both by magnetic and hydrodynamic forces in the environment.

Superparamagnetic Beads

The second simulation utilizes COMPEL 7.9μm diameter superparamagnetic beads. These beads were diluted in a 1% bovine serum albumin (BSA) solution to reduce non-specific binding between the beads and the substrate surface. The dilution had a very low concentration of beads to prevent bead-bead interactions induced by magnetic attraction, which often lead to beads sticking together to form doublets.

Tunable Magnetic Platform

The magnetic platform used to conduct experiments on the bacteria consists of a set of Helmholtz coils. Two pairs of Helmholtz coils give uniform global fields in the x and y directions, and the sample sits within another Helmholtz coil, producing a uniform z field, as

shown in Figure 1. The current in the coils is tunable between 10 and 100 Oe in each of the three dimensions, allowing a uniform magnetic field to be generated and manipulated to point in any direction. This field controls of the swimming direction of all the bacteria in a sample at once.

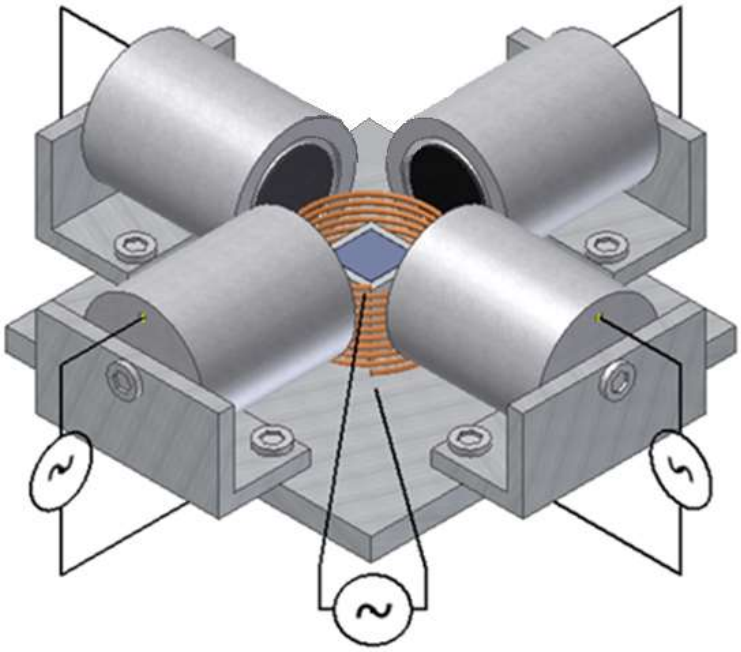

Figure 1: Helmholtz coil setup. Two opposite pairs of Helmholtz coils give the x and y uniform fields, and the sample (in blue) sits in the middle of another Helmholtz coil producing the out of plane z field.

Nonparallel Motion of AMB-1 by Hydrodynamic Veering

Independent control of individual bacteria is achieved using special configurations of the magnetic field, which can result in nonparallel motions that vary based on specific cell characteristics. One such configuration concerns cells tilted near a surface [32]. A global magnetic field with a nonzero component perpendicular to the surface of the sample causes the bacteria to swim towards either the top or bottom surface of the sample, depending on the cell's polarity, or choice of swimming direction along its constrained, one dimensional motion. For

these experiments, the cells swimming at the bottom surface were studied. Tilting this z field by using a nonzero in-plane (x or y) field component tilts the cells relative to the surface. The tilted cell's body rotates closer to the interface between the bulk fluid and the silicon surface and therefore experiences a greater resistive force than its counterrotating flagellum, which is further from the surface in this tilted orientation. This is shown in Figure 2a. The resulting rolling forces on the cell body and the flagellum are in opposite directions, but the magnitude of the rolling force on the cell body is greater, giving the cell a nonzero net rolling force.

Differentiated Swimming Direction

This rolling force causes the cell to diverge from the in-plane component of the global field by some characteristic "veer" angle, labelled φ in Figure 2b. The veer angle depends on the

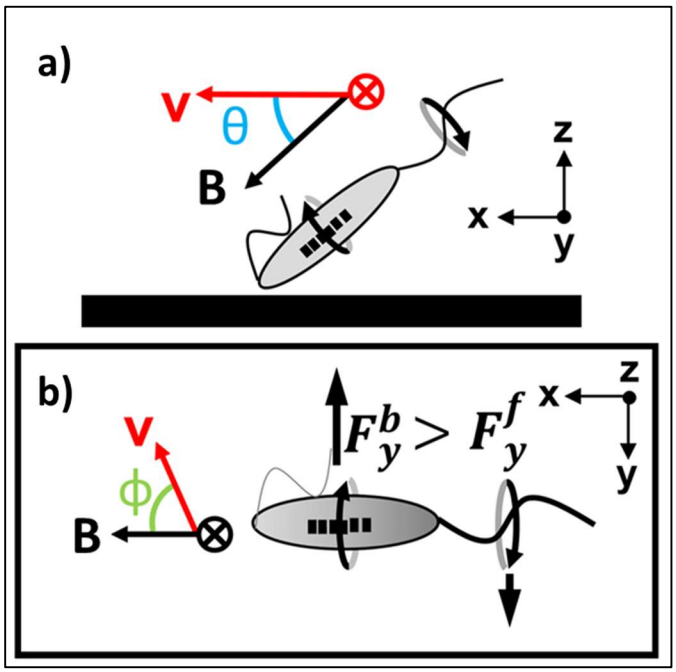

Figure 2: Diagram of an AMB-1 bacterium tilted near the surface. a) Side view, showing magnetic field, B, tilted at angle θ relative to the surface plane with cell's in-plane velocity v, and b) ariel view, showing the direction of the in-plane component of the magnetic field B and the resulting direction of the cell's velocity, diverging by angle φ. The rolling force on the body, F_y^b, is larger than that on the flagellum, F_y^f, giving rise to net rolling force

degree of the tilt angle, θ, as the difference between the proximity of flagellum and cell body to the surface is altered. The veer angle also varies from cell to cell, even at constant θ. Though this is not well characterized, this may be due to differences in shape of the cell body, both in length of the long axis or the tightness of the helical coils, or pitch, in the spirochete body. It may also be related to the elasticity or length of the active, rotating flagellum, the power output of the flagellum, or the inherent magnetic moment of the cell.

Differentiated Swimming Speed

The swimming speed also varies from cell to cell, and is similarly dependent on the degree of tilt relative to the surface, θ. As the cell is tilted towards the surface, a component of its velocity it pointed perpendicular to the surface, driving the cell into the surface, and no longer contributing to the cell's motion in the xy-plane. The component of the cell's velocity pointed in-plane decreases as θ increases. Accordingly, the recorded velocity as seen from directly above the sample decreases as θ increases. The veer angle and velocity were recorded for four cells as the global field was tilted incrementally between 10º and 80º in 10º steps. The magnitude of the field remained constant at 50 Oe. The results of this data are given in Figure 3.

Nonparallel Motion of Beads by Rotophoresis

The simulation has also been adapted for use of superparamagnetic beads as the robotic agents. These beads also exhibit nonparallel motion in response to a global control signal, where now the control signal is a rotating magnetic field with a specific frequency and axis of rotation. They exhibit differentiated motion only in their speed, rather than their orientation or direction of motion. Similarly to the bacteria, the nonparallel motion occurs as a result of mismatching drag

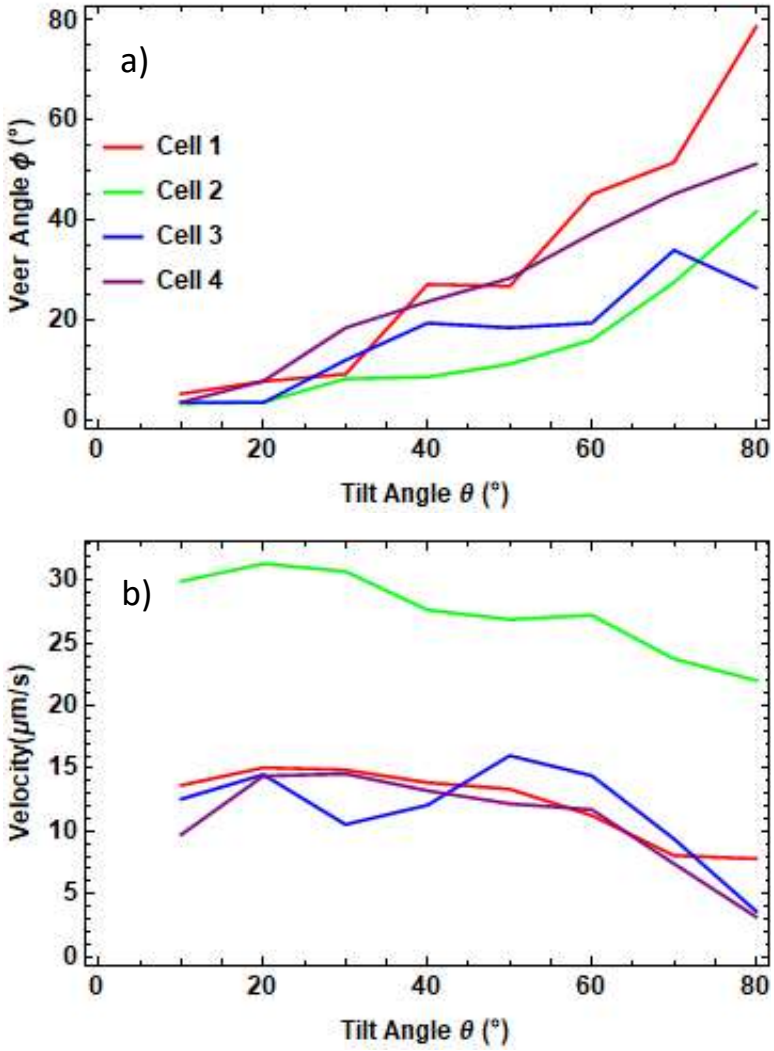

Figure 3: a) Veer angles and b) in plane velocities of four cells over set of global field tilt angles. Note that while the velocities of cells 1, 3, and 4 are fairly similar, their veer angles vary significantly, such that the overall motion is nonparallel.

forces of beads undergoing rotophoresis, or transporting via rolling, relative to a surface. The beads have an internal magnetic moment, which when out of alignment with the external magnetic field provides a torque. This torque rotates the bead to realign with the global field.

Frequency Dependence

The translational speed of each bead is dependent on the frequency of the rotating field, and the response also varies depending on the individual bead [33], [34]. In general, if the

external field is rotating slowly (out of plane), the bead's magnetic moment is constantly attempting to align with the field, and this applies a continuous torque on the bead, rotating the bead. In the bulk fluid, this means the bead rotates in place, but near a surface, we again encounter a mismatch in drag between the side of the bead closer to the surface and the side furthest from the surface. Thus the bead rolls along the surface along the global field's axis of rotation (in the xy-plane) with some velocity which is linearly dependent on the frequency of rotation of the global field at low frequencies. For beads of the same size this means we get roughly parallel motion, where each bead moves in the same direction with about the same velocity.

Critical Frequency

However, each bead has a critical frequency above which this linear relationship between bead velocity and external field rotation frequency breaks down. At frequencies higher than this critical frequency, the beads exhibit a nonparallel response as the transverse velocities decrease nonlinearly. This breakdown frequency occurs because at higher frequencies of external field rotation, the bead's physical rotation to align with the field cannot keep up. Eventually this means the phase lag becomes so great that the beads must rotate backwards to align with the field, slowing their average rolling speed. This breakdown frequency is also dependent on the individual beads, as each has a slightly different breakdown frequency due to differences in shape and in the magnitude of the magnetic moment, both of which vary due to manufacturing error. Figure 4 shows the relationship between the rotation frequency of the external field and the bead's translational speed for the three beads later used in simulation.

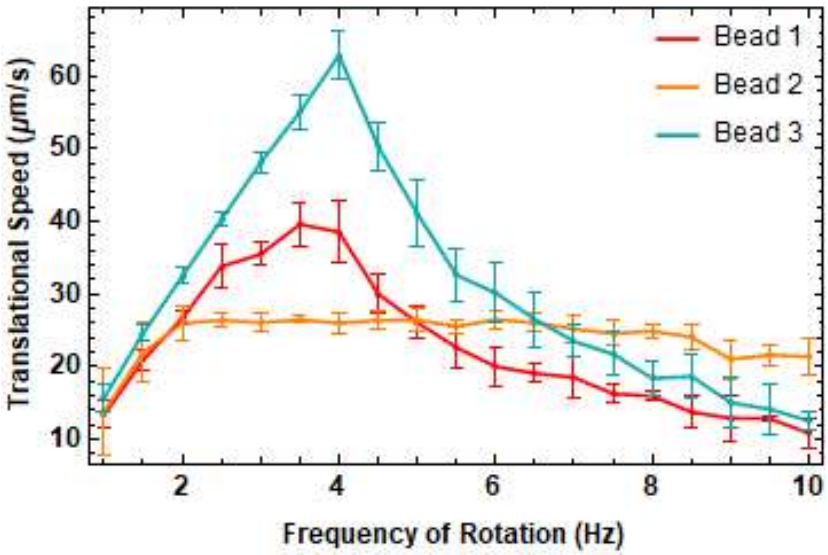

Figure 4: Bead translational speed vs. frequency of external magnetic field rotation. The frequencies from 0.5 to 10 Hz in 0.5 Hz increments. Each point is an average of six separate bead runs, and the error bars are the standard deviation of these six trials. For low frequencies, note the responses of the beads are linear and roughly equal. At higher frequencies each bead displays a different critical frequency where the translational speed peaks and then drops off nonlinearly. Above the first bead's critical frequency (Bead 2 and roughly 2 Hz) the speeds are significantly differentiated between beads.

Field Strength Dependence

The strength of the external rotating field also affects both the bead velocities and the value of the critical frequency for each bead. In general, for a single bead, if the field strength is lowered, the average speed is decreased, as is the critical frequency. Since the nonparallel motion of the beads hinge on this critical frequency, the strength of the magnetic field gives another parameter which can be tuned to generate linearly independent control signals. In the simulations of beads shown later, the control signals were defined using only the frequency and axis of rotation, but likely more beads could be controlled by adding in the field strength parameter, which we hope to do in future experiments.

Algorithm and Linear Algebra

Control Signals and Linear Independence

The following section details the algorithm used to obtain a set of "control signals" necessary to simultaneously transport multiple agents all subject to the same external fields. In the case of our experiments, a "control signal", denoted ψ, is a specific, global magnetic field applied across the entire sample. For bacteria, this is a static field tilted at a specific angle relative to the surface, but for the beads this is a uniform, rotating magnetic field with a specific frequency of rotation. In general, any tunable uniform field can be used as a control signal as long as it satisfies a few properties, as is discussed later in this section. As will also be shown later, the goal is to find a sufficient number of linearly independent control signals. This will ensure that there exists a specific set of times which specify the duration for which to apply each control signal, giving a complete description of the necessary field sequence to carry out the desired motion.

Configuration Space and Response Matrix

The algorithm relies heavily on the idea that the positions of N agents can be represented as a single point in a 2N-dimenional space. This position is written as a $2N \times 1$ vector with each agent's x and y positions making up the elements of the vector, shown transposed below as a row matrix for convenience, though in the algorithm is it used as a column matrix.

$$[x_1 \quad y_1 \quad x_2 \quad y_2 \quad \cdots \quad x_N \quad y_N]^T$$

A response vector, denoted **Ξ**, can be generated for each linearly independent control signal by defining the change in the agent's position in this 2N-dimensional space per unit time while subject to the given control signal. This requires data on all the agent's individual

displacements under the influence of that control signal, so that the response vector can be built from the individual changes in x and y of each agents. A single response vector is given below, showing that if the i^{th} control signal, ψ_i, is applied, the total response of the configuration of all agents is given by the individual displacements of each agents per unit time, notated using u as the speed in the x direction, and v as the speed in the y direction.

$$\boldsymbol{\Xi_i} = [u_1^i \quad v_1^i \quad u_2^i \quad v_2^i \quad \cdots \quad u_n^i \quad v_n^i]^T$$

Given a set of control signals, there will be a corresponding set of response vectors which can be combined into a response matrix, where each column of the matrix is a separate response vector.

Linear Combinations

If the response vectors corresponding to the control signals are linearly independent, and if the number of response vectors is equal to the number of dimensions in the space of interest, then a linear combination of those response vectors can be used to reach any point in the space. For a 2-dimensional space, unit vectors $\hat{x}$ and $\hat{y}$ are two linearly independent vectors, and from these we can define any point in the 2-dimensional space using the form $a\hat{x} + b\hat{y}$ to get a unique point (a, b). Similarly, in the 2N-dimensional configuration space, if we have 2N linearly independent response vectors, any specific configuration of all N agents, represented as a point in the configuration space, can be reached using a linear combination of the response vectors.

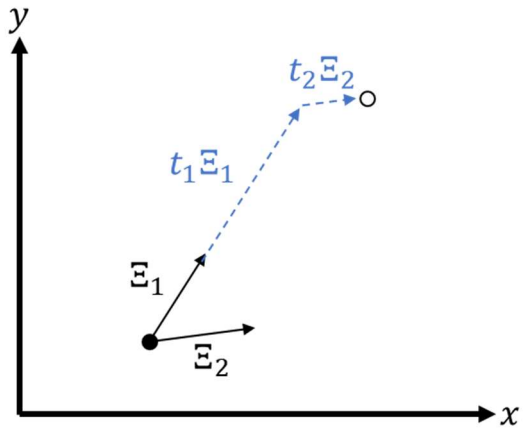

Figure 5: Diagram of linear combination of response vectors in 2N-dimensional configuration space, specifically here N=1, implying a single agent. The response vectors $\boldsymbol{\Xi_1}$ and $\boldsymbol{\Xi_2}$ give two response vectors for displacement per one unit time. These "unit" vectors are scaled by times t_1 and t_2 and added to reach the arbitrary final point (outlined dot) from the given initial point (black dot). This is equivalent to applying the corresponding control signals sequentially for these respective times. Extension to N>1 requires more response vectors, but the process of scaling and adding the vectors remains the same.

$$t_1\boldsymbol{\Xi_1} + t_2\boldsymbol{\Xi_2} + \cdots + t_{(2N-1)}\boldsymbol{\Xi_{(2N-1)}} + t_{2N}\boldsymbol{\Xi_{2N}}$$

The scalar coefficients give the units of time for which the control signal corresponding to that response vector should be applied. For example, since $\boldsymbol{\Xi_1}$ corresponds to a specific magnetic field, t_1 specifies how long that field should be applied to the agents. Determining all of the times t_i gives how long each of the control signals should be applied, in sequence, to carry out the desired motion, and thus fully specifies the necessary sequence of fields.

Calculating the Scalar Coefficients

To figure out the times, a matrix equation is needed using both the response matrix and the desired final configuration. The final configuration is denoted $\boldsymbol{X}$ and is represented by a $2N \times 1$ column vector as presented earlier. It is given by the desired (x_f, y_f) ending positions of each agent, minus the starting positions (x_i, y_i). This simply treats the starting configuration as the origin of the configuration space. The scalar coefficients also form a $2N \times 1$ column vector, and the matrix equation is shown below.

$$\begin{bmatrix} u_1^1 & u_1^2 & \cdots & u_1^{2N} \\ v_1^1 & v_1^2 & & v_1^{2N} \\ u_2^1 & u_2^2 & & u_2^{2N} \\ v_2^1 & v_2^2 & \ddots & v_2^{2N} \\ \vdots & \vdots & & \vdots \\ u_n^1 & u_n^2 & \cdots & u_n^{2N} \\ v_n^1 & v_n^2 & & v_n^{2N} \end{bmatrix}_{2N\times 2N} \begin{bmatrix} t_1 \\ t_2 \\ \\ \\ \vdots \\ \\ t_{2N} \end{bmatrix}_{2N\times 1} = \begin{bmatrix} x_f^1 - x_i^1 \\ y_f^1 - y_i^1 \\ x_f^2 - x_i^2 \\ y_f^2 - y_i^2 \\ \vdots \\ x_f^n - x_i^n \\ y_f^n - y_i^n \end{bmatrix}_{2N\times 1}$$

We can more compactly represent this equation by letting the response matrix be notated as $\boldsymbol{A}$, the time vector as $\boldsymbol{\tau}$, and the final configuration as $\boldsymbol{X}$.

$$\boldsymbol{A\,\tau = X}$$

From this equation, we see that if the response matrix is a square $2N \times 2N$ matrix, then it is easy to solve for τ using the inverse of $\boldsymbol{A}$, since a matrix multiplied by its inverse gives the identity matrix.

$$\boldsymbol{A^{-1}\,A\,\tau = A^{-1}\,X = \tau}$$

Thus, with 2N linearly independent response vectors, we can solve for the complete sequence needed to transport all agents simultaneously to any arbitrary final configuration.

Necessary Properties of Control Signals

For systems other than bacteria and beads under magnetic fields, this method will also work, as long as there exists a set of control signals which satisfy a few properties. Namely, unique control signals must exist which produce reliable responses in the agents, such that each time they are applied the same responses are observed. The signals should also have an "inverse", or a corresponding signal which produces the opposite response, as this is necessary later to give a physical significance to negative times. For a tilted magnetic field, rotating the in-

plane component of the field by 180º exactly reverses the bacteria's motion. For the rotating magnetic field, flipping the axis of rotation by 180º similarly reverses the bead's motion. Finally, the control signals need to produce nonlinear (or nonparallel) responses in the agents. This means that a linear change in the control signal should not produce a similarly linear response change. This is the most important consideration, as otherwise the only possible motions would be to translate the entire system in the xy-plane, rather than actually changing the configuration of agents relative to each other.

Consolidating Control Signals

In practice, having fewer separate control signals is beneficial to save time computationally. This can be accomplished by consolidating control signals which differ only by their in-plane direction. For example, we will see that for bacteria and beads, two separate control signals are obtained by rotating the same magnetic field (i.e. same tilt angle or same frequency, respectively) by 90º. These "conjugate signals" are related by a rotation and thus linearly independent and can be combined. If signal ψ is applied for time t, and its "conjugate"

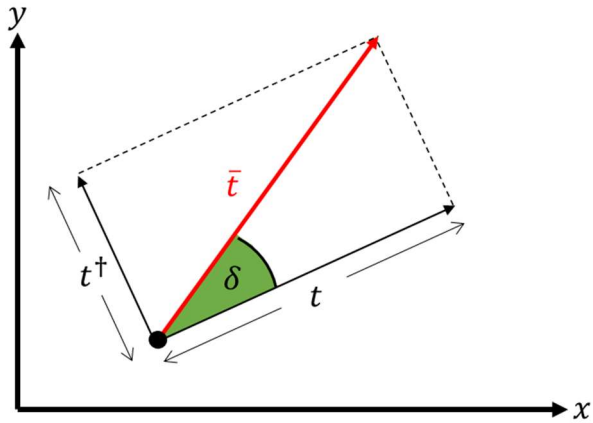

Figure 6: Two "conjugate" control signals, ψ and $\psi^\dagger$, corresponding to vectors in 2N-dimensional space with lengths t and $t^\dagger$ respectively can be combined to form a new, consolidated vector with length $\bar{t}$ and angle δ. This is equivalent to applying both conjugate signals sequentially but saves time since $\bar{t} < t + t^\dagger$.

signal $\psi^{\dagger}$ is applied for time $t^{\dagger}$, then the combined signal $\bar{\psi}$ has direction $\delta = \tan^{-1}\left(\frac{t^{\dagger}}{t}\right)$ and length $\bar{t} = \sqrt{(t^{\dagger})^2 + (t)^2}$, as shown in Figure 6. This new signal is exactly equivalent to applying the two original signals but is easier to implement both in simulation and in practice.

Strategy for Using Fewer Response Vectors

In general, as the number of agents increases, obtaining 2N distinct control signals (and therefore 2N response vectors) gets increasingly more difficult. Consider the bacteria system, where the total number of control signals depends on the number of distinct tilt angles. As the number of agents increases more distinct tilt angles are needed, and eventually there is a point where tilt angles are too close together to give significant differentiation of motion. In this case it is necessary to find ways to minimize the amount of control signals required for the algorithm. If we have less than 2N response vectors, A is no longer a square matrix, and therefore is not invertible. However, multiplying A by its transpose, A^T, does give a square matrix, which can then be inverted. Then the matrix given by $A^T\,A$ acts exactly like A did previously when there were 2N response vectors.

$$(A^T\,A)^{-1}\,(A^T\,A)\,\tau = \,(A^T\,A)^{-1}\,X = \,\tau$$

If this equation can be solved such that the column vector τ exists and is made up of real values (i.e. real times), then the final configuration X can be reached using this set of fewer than 2N response vectors. This idea is clarified using a 2-dimensional example. If the chosen final point happened to lie along the x axis, then that point could be reached using a single unit vector, $\hat{x}$, even though reaching an arbitrary point still requires a combination of both $\hat{x}$ and $\hat{y}$. This same process is what allows some configurations to be reached using a smaller set of vectors in 2N-dimensional space.

The python code implements the algorithm using this method rather than assuming that the response matrix is square, although for the purposes of the simulation, 2N response vectors were available. This code is given as an Appendix.

Path Planning

Executing the sequence by applying the full duration of each control signal before switching to the next control signal presents an issue. This method generally leads to the agents travelling long distances in one direction, which can be problematic in a real-world setup with a limited frame of view. To avoid losing view of the agents of interest, a function called "path planning" separates the sequence into smaller pieces, as illustrated in Figure 7. A control signal which must be applied for a relatively long time can instead be applied multiple times for shorter intervals, as long as the sum of these shorter intervals equals the original time. This means that the whole sequence can be strategically applied piecewise to keep the agents from leaving the frame during their motion. The "path planning" function works by determining when an agent reaches the frame's boundary, and at that point switches the control signal to the next in the

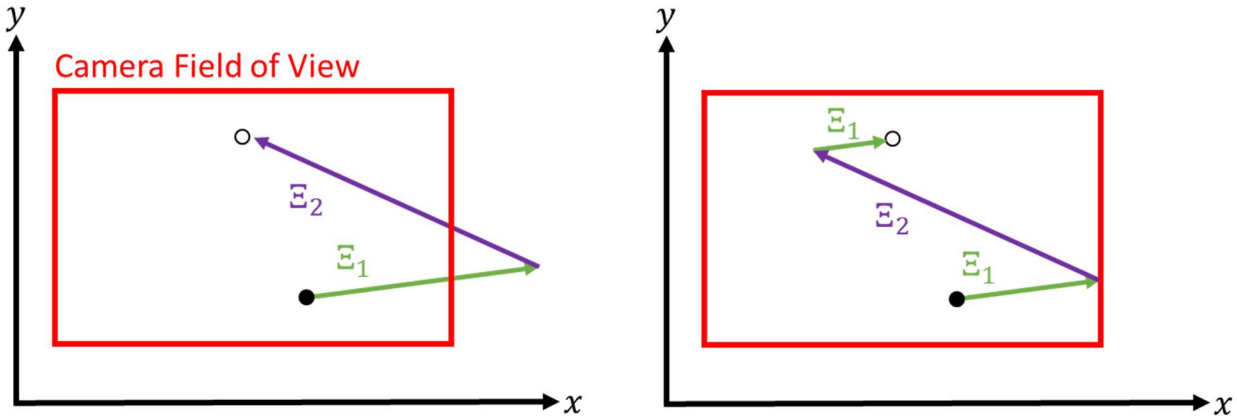

Figure 7: Illustration of path planning for a single agent. The left shows the sequential application of two control signals, where the agent moves outside the camera's field of view. By breaking up the first control signal into two smaller pieces and applying them separately the agent can be kept in the field of view and still reach the desired final position.

sequence. If this control signal fails to bring the agent back within the frame, the signal is switched to the next in the sequence, and so on until a control signal is reached which brings the agent back in frame. This is a fairly rudimentary scheme, and improvements to this method of keeping agents within view are discussed in the future work section.

Bacterial Simulation

For the bacteria, each element in the response vector is the change in x and y of a specific bacterium if a field with a given tilt angle is applied for one unit of time. This value can be found using basic trigonometry given velocity and cell orientation (swimming direction) relative to the positive y direction. This is given completely by the veer angle and velocity data.

Determining Control Signals

Each control signal for the bacteria is a static magnetic field tilted relative to the surface at some tilt angle θ, which is characterized both by the specific tilt angle and the direction of the

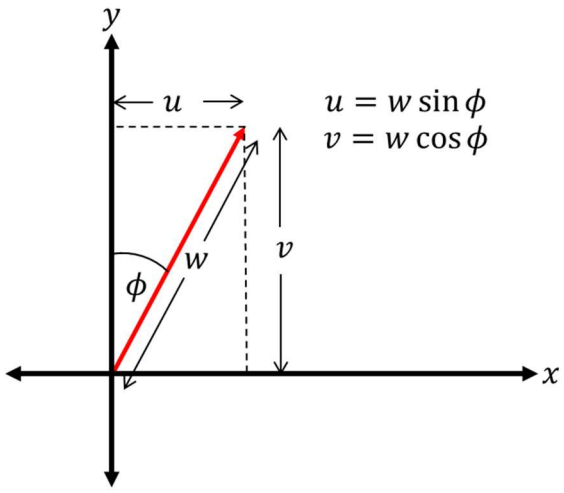

Figure 8: Calculation of response vector elements given veer angle ϕ and total in-plane cell velocity w. Per unit time, the in-plane displacement is equal to the velocity. Simple trigonometry gives the x and y components (u and v respectively) of w.

field's in-plane component. There are already N response vectors if we take N magnetic fields with distinct tilt angles and all with the in-plane component of the global field along the positive y direction. These response vectors are calculated from the bacterial motion using simple trigonometry (Figure 8). These control signals are taken to be effectively linearly independent. To get the other N response vectors necessary, the same magnetic fields are rotated by 90º, such that they have the same tilt angles as the first N control signals, but now the in-plane component of the field points along the positive x direction. This rotation gives another set of effectively linearly independent control signals. Furthermore, this set is directionally orthogonal to the first set, and therefore the two sets are linearly independent of each other. This gives a total of 2N linearly independent response vectors, and from this we can compute the duration to apply each of the 2N fields using code which implements the algorithm from the previous section. Pairs of control signals with the same tilt angle are consolidated according to the method previously outlined.

Simulation Results

The result of the simulation is that four distinct bacteria can be moved to specified ending positions from a given initial configuration. In Figure 9, four cells are initially positioned along a vertical line, and are moved to occupy the four corners of a square. The still frames of this animation show the progression of steps.

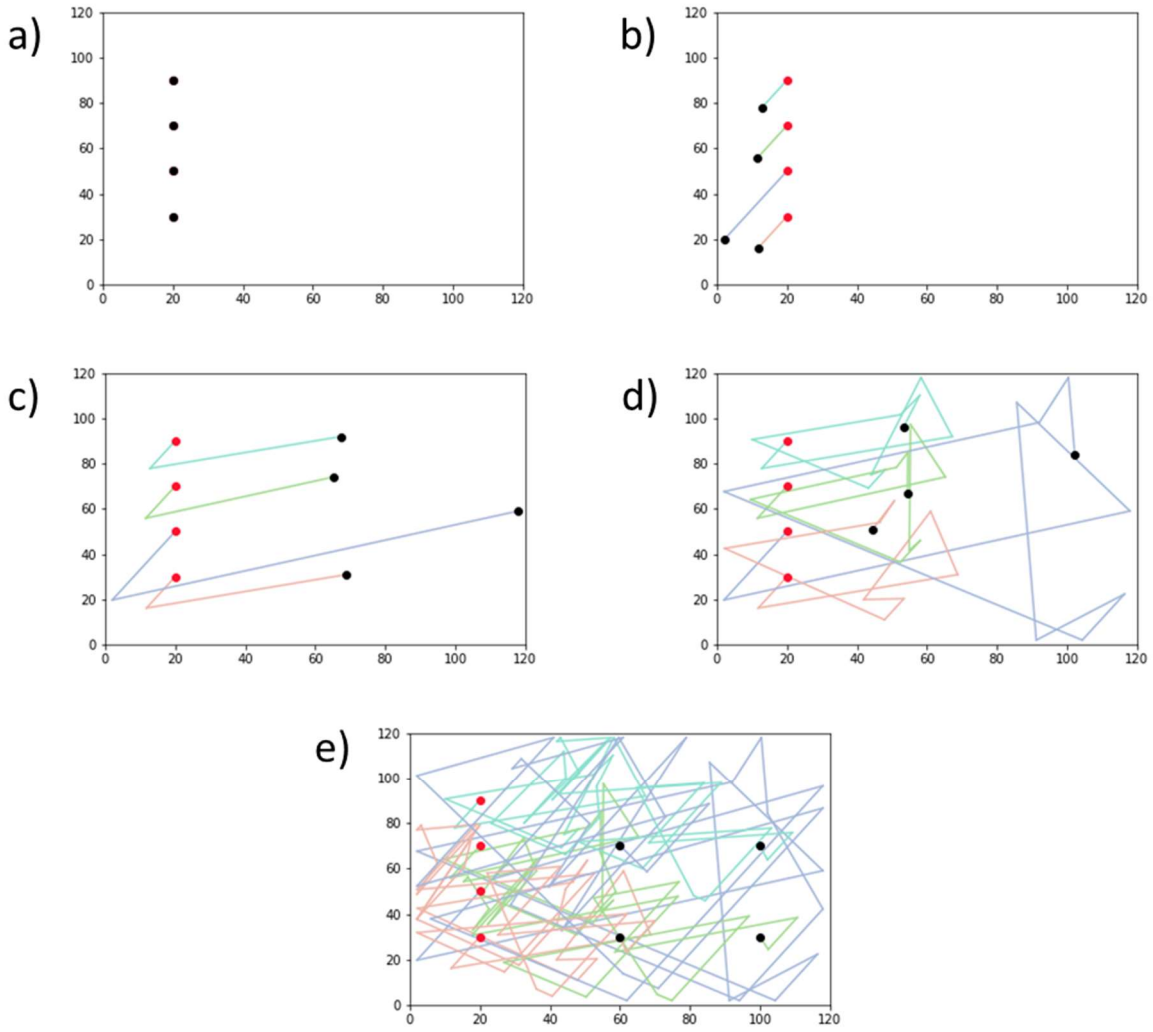

Figure 9: Simulation frames of the AMB-1 motion for four cells moving simultaneously. All distances are given in μm. In a) the bacteria are initially arranged along a vertical line. b) shows the result of applying the first control signal, where the initial positions are in red and the current positions are in black. The lines show the trajectory of each cell. The results after c) two control signals and d) ten control signals are also given. e) gives the final result, where the cells are arranged in a square.

Bead Simulation

For the beads, determining the response vectors resulting from a magnetic field with a given frequency and axis of rotation is simple, since the only difference between beads is in velocity. Therefore, the response of each bead is simply the component of its velocity along the

direction of the axis of rotation for the given field (i.e., for a field with axis of rotation along the y axis, each bead's response is simply $u = 0, v = w$, where w is the bead's translational speed).

Determining Control Signals

We again obtain N control signals using magnetic fields with the same axis of rotation (along the positive y direction), but with distinct frequencies of rotation. The second set of N control signals rotates the axis of rotation so that it lies along the positive x direction. The signals with the same frequency are consolidated so that N total control signals remain.

Simulation Results

The algorithm is identical for the beads' motion, and a similar set of stills from the animation of the bead motion is given in Figure 10. In this simulation, three beads were aligned along a horizontal line at the bottom of the frame, and then moved to form a triangle higher in the field of view. The simulation was unable to develop an animation demonstrating control of four beads. For this early version of the algorithm, bead data presents too little variability to reliably transport four beads. Improvements both to the algorithm and to the bead dataset are discussed as part of the future work section.

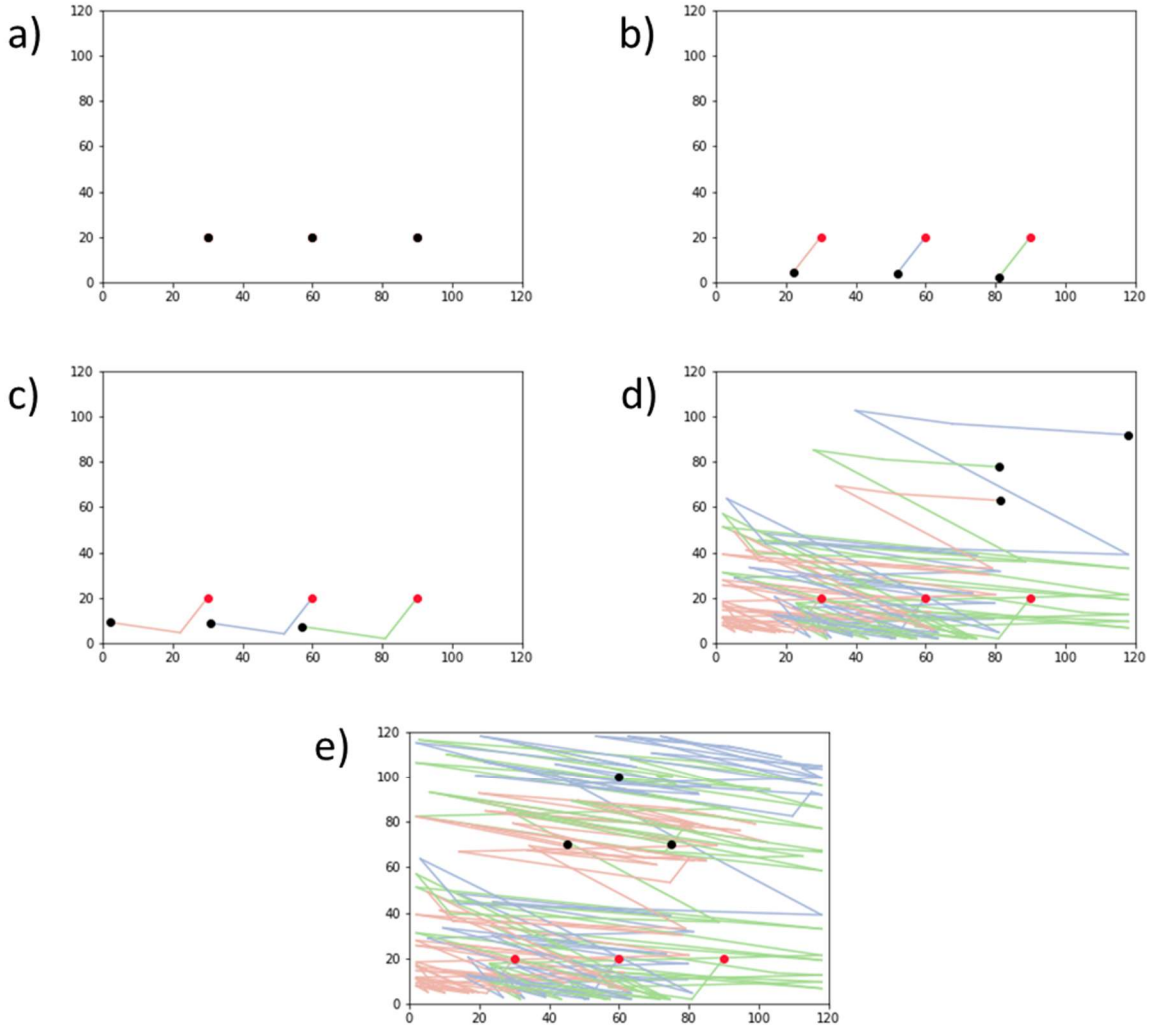

Figure 10: Simulation frames of the bead motion for three beads moving simultaneously. All distances are given in μm. In a) the beads are initially arranged along a horizontal line at the bottom of the field of view. b) shows the result of applying the first control signal, where the initial positions are in red and the current positions are in black. The lines show the trajectory of each cell. The results after c) two control signals and d) forty control signals are also given. e) gives the final result, where the beads are arranged in a triangle.

Noise Distribution

While there is, unfortunately, no noise measurements on the four AMB-1 cells used in the simulation, there is data on the positional uncertainty of the beads. For each bead, six separate runs with the same frequency and bead strength were taken such that we get a standard deviation

of the velocity for each control signal. This standard deviation can be factored into the simulation to figure out how close the algorithm truly gets the bead to their final configuration when the system is noisy. This will be an important consideration when implementing the algorithm in real time, when stochasticity in the positions and velocities in unavoidable. Figure ## shows the final positions achieved for 50 iterations, where each step factors in the average noise in the beads' speed. This spread suggests that the algorithm does need to be improved to account for noise before it is ready to be applied to a real system. Ideas for dealing with noisy systems is discussed in the following section.

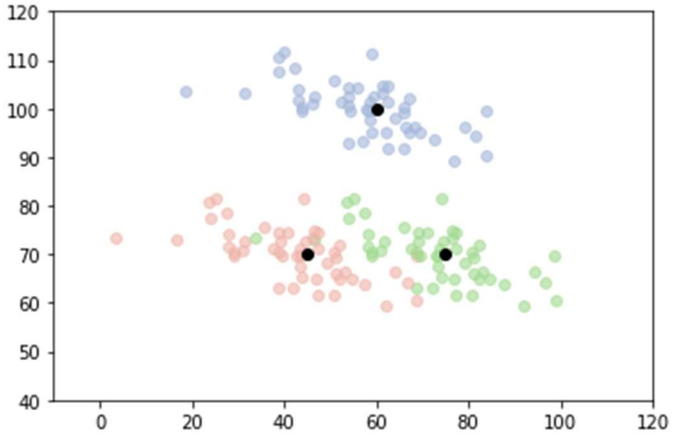

Figure 11: Distribution of final positions due to noise for the bead simulation. The distribution of each agent's final position is given by the separate colors, and the desired final positions are given by the black points. All distances are given in μm.

Future Work

Future work will largely focus on both improving the algorithm and engineering a computer vision system such that the computer can track the agents in real time, determine the control sequence, and apply the required magnetic fields.

For beads in particular, adding in another parameter to the control signals, the strength of the external field, should offer improvements on the simulation. The added control signals give a wider range of available response vectors, each with varying levels of differentiation in the bead motion. While theoretically the algorithm should be able to find a sequence to transport any number of beads, the number of steps to complete the motion is prohibitively large. This greater variability in differentiation, should help reduce the number of steps necessary to complete the motion, as it introduces fewer constraints to the possible bead motions. This, along with the other improvements to the algorithm suggested below, should help to increase the number of beads we are able to control in simulation.

To improve the algorithm, the first focus is on creating a more effective path planning function to keep the agents in the camera's frame of view. A cost function would be more effective than switching control signals in keeping the bacteria in the field of view. The current system of switching control signals often requires making many small steps, where a control signal is applied for potentially fractions of a second. This is difficult to implement in real life, as generating a stable magnetic field takes a finite amount of time. It is preferential then to minimize the number of times that the control signal needs to be switched. Building a cost function based on available control signals and proximity to an edge or a corner in the frame of view may help keep the number of steps in the sequence to a minimum.

Secondly, we will need to factor in obstacle avoidance. There tend to be areas which should be avoided in the real system, such as an agent or other object stuck to the surface. Additionally, when agents get too close to each other, they interact in a way which will affect their trajectories. Bacteria too close to each other will experience hydrodynamic attraction and couple, and beads which are too close experience magnetic attraction and will form a doublet.

This doublet has a significantly different response to the global field than a single bead, and thus the response vectors change, something which we currently cannot account for in the simulation. Therefore other moving agents for now should be considered obstacles to be avoided.

The number of distinct control signals needed scales with the number of agents to control at once. Since 2N linearly independent control signals are needed for N agents, this can be problematic for higher numbers of agents. Therefore, it would be helpful to add one control signal at a time to the total set of control signals and perform checks after each addition to see whether the resulting set of response vectors are already sufficient to reach the specified final configuration. This requires determining a control signal and quantifying the agents' response in real time, thereby generating the corresponding response vector. As each response vector is determined, it may be added to the response matrix, A. A is rectangular if we have less than 2N response vectors, the procedure outline in the linear algebra section of the paper can be used to determine whether there is a set of real times τ which solves the matrix equation. If so, this set of response vectors already reaches the specified final configuration and we can stop adding control signals. If not, then more control signals need to be added. Determining one response vector at a time, adding it as a column in A, and testing if the set is sufficient, would help reduce the number of necessary control signals.

Finally, we would like to consider noise in the system, and find ways to correct for this stochasticity so that the final configuration is reached with some accuracy. One idea for this is to check the positions of the agents at specified checkpoints, defined either by time intervals or after a certain number of steps. If the agents' positions deviate from the expected position over a certain tolerance, the actual positions at this point may act as a new initial configuration. From this new initial configuration, a new sequence could be obtained using the original final

configuration. Another option is to use the expected positions as a new final configuration, thereby making an adjustment to get the agents back to their expected position and then continuing with the original control sequence. Either of these methods should correct for noise as the motion proceeds.

To implement this procedure in real time, a computer vision system is necessary to track the agents. Additionally, the computer needs to be able to obtain the sequence using the algorithm and apply the magnetic fields to the sample. First, the desired final configuration should be specified. The agents identified in the field of view should be subjected to one of the possible control signals, and the resulting motions of each agent should be recorded, as this constitutes their response. This step should be completed, adding one control signal/response at a time, until the final configuration can be reached. The resulting magnetic field sequence should be applied according to the path planning function to keep the agents in the field of view, and periodic checks should be performed to correct for deviation due to noise. When the agents Euclidean distance between the current position and desired final position (in the 2N-dimensional space) is within some predetermined tolerance, the motion is complete.

Acknowledgements

This work was supported by a National Science Foundation grant no. 1710598. I would also like to thank Hiran Wijesinghe for his contributions to the development of the mathematical framework, Prof. Ratnasingham Sooryakumar for his support, and Prof. Eric Mumper for providing the AMB-1 cultures.

References

[1] David Cappelleri, Dimitrios Efthymiou, Ashesh Goswami, Nikolaos Vitoroulis, and Michael Zavlanos. Towards Mobile Microrobot Swarms for Additive Micromanufacturing. International Journal of Advanced Robotic Systems, 11(9):150, September 2014. http://journals.sagepub.com/doi/10.5772/58985.

[2] Metin Sitti, Hakan Ceylan, Wenqi Hu, Joshua Giltinan, Mehmet Turan, Sehyuk Yim, and Eric Diller. Biomedical Applications of Untethered Mobile Milli/Microrobots. Proceedings of the IEEE, 103(2):205–224, February 2015. http://ieeexplore.ieee.org/document/7067029/.

[3] Bradley J. Nelson, Ioannis K. Kaliakatsos, and Jake J. Abbott. Microrobots for Minimally Invasive Medicine. Annual Review of Biomedical Engineering, 12(1):55–85, July 2010. http://www.annualreviews.org/doi/10.1146/annurev-bioeng-010510-103409.

[4] Peyer KE, Zhang L, Nelson BJ. Bio-inspired magnetic swimming microrobots for biomedical applications. Nanoscale. 2013 Feb 21;5(4):1259-72. doi: 10.1039/c2nr32554c. PMID: 23165991.

[5] Lauback, S., Mattioli, K.R., Marras, A.E. et al. Real-time magnetic actuation of DNA nanodevices via modular integration with stiff micro-levers. Nat Commun 9, 1446 (2018). https://doi.org/10.1038/s41467-018-03601-5

[6] Gao, Wei & Kagan, Daniel & Pak, On Shun & Clawson, Corbin & Campuzano, Susana & Chuluun-Erdene, Erdembileg & Shipton, Erik & Fullerton, Eric & Zhang, Liangfang & Lauga, Eric & Wang, Joseph. (2012). Cargo-Towing Fuel-Free Magnetic Nanoswimmers for Targeted Drug Delivery. Small (Weinheim an der Bergstrasse, Germany). 8. 460-7.10.1002/smll.201101909.

[7] Felfoul, O., Mohammadi, M., Taherkhani, S. et al. Magneto-aerotactic bacteria deliver drug-containing nanoliposomes to tumour hypoxic regions. Nature Nanotech 11, 941–947 (2016). https://doi.org/10.1038/nnano.2016.137

[8] Singh, Ajay & Hosseinidoust, Zeinab & Yaşa, Öncay & Sitti, Metin. (2017). Microemulsion-Based Soft Bacteria-Driven Microswimmers for Active Cargo Delivery. ACS Nano. 11.10.1021/acsnano.7b02082.

[9] Sylvain Martel. Magnetic Navigation Control of Microagents in the Vascular Network: Challenges and Strategies for Endovascular Magnetic Navigation Control of Microscale Drug Delivery Carriers. IEEE Control Systems, 33(6):119–134, December 2013. https://ieeexplore.ieee.org/document/6615617/.

[10] Ashis Gopal Banerjee, Sagar Chowdhury, Wolfgang Losert, and Satyandra K. Gupta. Survey on indirect optical manipulation of cells, nucleic acids, and motor proteins. Journal of Biomedical Optics, 16(5):051302, 2011. http://biomedicaloptics.spiedigitallibrary.org/article.aspx?doi=10.1117/1.3579200.

[11] Sagar Chowdhury, Wuming Jing, and David J. Cappelleri. Controlling multiple microrobots: Recent progress and future challenges. Journal of Micro-Bio Robotics, 10(1-4):1–11, October 2015. http://link.springer.com/10.1007/s12213-015-0083-6.

[12] Rahman, M.A., Cheng, J., Wang, Z. et al. Cooperative Micromanipulation Using the Independent Actuation of Fifty Microrobots in Parallel. Sci Rep 7, 3278 (2017). https://doi.org/10.1038/s41598-017-03525-y

[13] Montenegro-Johnson, Thomas. (2018). Microtransformers: Controlled microscale navigation with flexible robots. Physical Review Fluids. 3. 10.1103/PhysRevFluids.3.062201.

[14] Hashimoto, Koichi & Takahashi, Kiyonori & Ogawa, Naoko & Oku, Hiromasa. (2006). Visual Feedback Control for a Cluster of Microorganisms. 2006 SICE-ICASE International Joint Conference. 10.1109/SICE.2006.314769.

[15] Charles R. P. Courtney, Christine E. M. Demore, Hongxiao Wu, Alon Grinenko, Paul D. Wilcox, Sandy Cochran, and Bruce W. Drinkwater , "Independent trapping and manipulation of microparticles using dexterous acoustic tweezers" , Applied Physics Letters 104, 154103 (2014) https://doi.org/10.1063/1.4870489

[16] Steager, Edward & Wong, Denise & Wang, Jeremy & Arora, Simran & Kumar, Vijay. (2017). Control of multiple microrobots with multiscale magnetic field superposition. 1-6. 10.1109/MARSS.2017.8001927.

[17] D. Wong, E. B. Steager and V. Kumar, "Independent Control of Identical Magnetic Robots in a Plane," in IEEE Robotics and Automation Letters, vol. 1, no. 1, pp. 554-561, Jan. 2016, doi: 10.1109/LRA.2016.2522999.

[18] Henighan, T., Chen, A., Vieira, G., Hauser, A. J., Yang, F. Y., Chalmers, J. J., & Sooryakumar, R. (2010). Manipulation of magnetically labeled and unlabeled cells with mobile magnetic traps. Biophysical journal, 98(3), 412–417. https://doi.org/10.1016/j.bpj.2009.10.036

[19] Vieira, Gregory & Chen, A. & Henighan, T. & Lucy, J. & Yang, Fair & Sooryakumar, R.. (2012). Transport of magnetic microparticles via tunable stationary magnetic traps in patterned wires. Physical Review B. 85. 174440-. 10.1103/PhysRevB.85.174440.

[20] Diller, E., Giltinan, J., & Sitti, M. (2013). Independent control of multiple magnetic microrobots in three dimensions. The International Journal of Robotics Research, 32(5), 614–631. https://doi.org/10.1177/0278364913483183

[21] Yan Ou, D. H. Kim, P. Kim, M. J. Kim and A. A. Julius, "Motion control of Tetrahymena pyriformis cells with artificial magnetotaxis: Model Predictive Control (MPC) approach,"

2012 IEEE International Conference on Robotics and Automation, Saint Paul, MN, USA, 2012, pp. 2492-2497, doi: 10.1109/ICRA.2012.6225015.

[22] Y. Ou, P. Kang, M. J. Kim and A. A. Julius, "Algorithms for simultaneous motion control of multiple T. pyriformis cells: Model predictive control and Particle Swarm Optimization," 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA, 2015, pp. 3507-3512, doi: 10.1109/ICRA.2015.7139684.

[23] A. Becker, Y. Ou, P. Kim, M. J. Kim and A. Julius, "Feedback control of many magnetized: Tetrahymena pyriformis cells by exploiting phase inhomogeneity," 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, 2013, pp. 3317-3323, doi:10.1109/IROS.2013.6696828.

[24] K. Das and D. Ghose, "Positional consensus in multi-agent systems using a broadcast control mechanism," 2009 American Control Conference, St. Louis, MO, USA, 2009, pp. 5731-5736, doi: 10.1109/ACC.2009.5160384.

[25] Floyd, Steven & Diller, Eric & Pawashe, Chytra & Sitti, Metin. (2011). Control methodologies for a heterogeneous group of untethered magnetic micro-robots. I. J. Robotic Res.. 30. 1553-1565. 10.1177/0278364911399525.

[26] Lidong Yang and Li Zhang. Motion Control in Magnetic Microrobotics: From Individual and Multiple Robots to Swarms. Annual Review of Control, Robotics, and Autonomous

Systems, 4(1):annurev–control–032720–104318, May 2021. https://www.annualreviews.org/doi/10.1146/annurev-control-032720-104318.

[27] Arne Schmidt, Victor M. Baez, Aaron T. Becker, and Sandor P. Fekete. Coordinated Particle Relocation Using Finite Static Friction With Boundary Walls. IEEE Robotics and Automation Letters, 5(2):985–992, April 2020. https://ieeexplore.ieee.org/document/8962023/.

[28] Shiva Shahrokhi, Jingang Shi, Benedict Isichei, and Aaron T. Becker. Exploiting Nonslip Wall Contacts to Position Two Particles Using the Same Control Input. IEEE Transactions on Robotics, 35(3):577–588, June 2019. https://ieeexplore.ieee.org/document/8638552/.

[29] Eric Diller, Steven Floyd, Chytra Pawashe, and Metin Sitti. Control of Multiple Heterogeneous Magnetic Microrobots in Two Dimensions on Nonspecialized Surfaces. IEEE Transactions on Robotics, 28(1):172–182, February 2012. http://ieeexplore.ieee.org/document/6056575/.

[30] Dominic R. Frutiger, Karl Vollmers, Bradley E. Kratochvil, and Bradley J. Nelson. Small, Fast, and Under Control: Wireless Resonant Magnetic Micro-agents. The International Journal of Robotics Research, 29(5):613–636, April 2010. http://journals.sagepub.com/doi/10.1177/0278364909353351.

[31] U Kei Cheang, Kyoungwoo Lee, Anak Agung Julius, and Min Jun Kim. Multiple robot drug delivery strategy through coordinated teams of microswimmers. Applied Physics Letters, 105(8):083705, August 2014. http://aip.scitation.org/doi/10.1063/1.4893695.

[32] Pierce CJ, Mumper E, Brown EE, Brangham JT, Lower BH, Lower SK, Yang FY, Sooryakumar R. Tuning bacterial hydrodynamics with magnetic fields. Phys Rev E. 2017 Jun;95(6-1):062612. doi: 10.1103/PhysRevE.95.062612. Epub 2017 Jun 30. PMID: 28709362.

[33] X.J.A. Janssen, A.J. Schellekens, K. van Ommering, L.J. van IJzendoorn, and M.W.J. Prins. Controlled torque on superparamagnetic beads for functional biosensors. Biosensors and Bioelectronics, 24(7):1937–1941, March 2009. https://doi.org/10.1016/j.bios.2008.09.024.

[34] C. Pease, H.S. Wijesinghe, J. Etheridge, C.J. Pierce, and R. Sooryakumar. Magnetic and hydrodynamic torques: Dynamics of superparamagnetic bead doublets. Journal of Magnetism and Magnetic Materials, 466:323–332, November 2018. https://doi.org/10.1016/j.jmmm.2018.07.014.

Appendix A

Python code which implements the described algorithm where input is starting and ending configurations specified by the user, and the output is the sequence of control signals and corresponding durations. This is separate from path planning. This uses python package numpy and linalg from scipy.

```
def get_sequence(start_config, end_config):

    cell_num = len(start_config)
    tilt_angles = [0,10,20,30,40,50,60,70,80]
    freqs = [0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0, 5.5, 6.0, 6.5, 7.0, 7.5, 8.0, 8.5, 9.0, 9.5]
    num_tilts = len(tilt_angles)
    num_freqs = len(freqs)
    if (data==cell_data):
        control = tilt_angles
        control_num = num_tilts
    else:
        control = freqs
        control_num = num_freqs
    velocities = []
    veer_angles = []
    for k in range(control_num):
        velocities.append(tilt_to_vd(control[k], data)[0])
        veer_angles.append(tilt_to_vd(control[k], data)[1]*(np.pi/180))
    response_vectors = []
    res_vec_control_signal = []
    for k in range(control_num):
        response_array = np.zeros([2*cell_num, 1])
        response_array_rotation = np.zeros([2*cell_num, 1])
        for j in range(cell_num):
            response_array[j] = velocities[k][j]*np.sin(veer_angles[k][j])
            response_array[j+cell_num] = velocities[k][j]*np.cos(veer_angles[k][j])
            response_array_rotation[j] = velocities[k][j]*np.cos(veer_angles[k][j])
            response_array_rotation[j+cell_num] = velocities[k][j]*np.sin(veer_angles[k][j])*(-1)
        response_vectors.append(response_array)
        response_vectors.append(response_array_rotation)
        res_vec_control_signal.append((0, control[k]))
        res_vec_control_signal.append((90, control[k]))
    start_config_rearrange = np.array([])
    end_config_rearrange = np.array([])
    for i in range(2):
```

```
    for j in range(cell_num):
        start_config_rearrange = np.append(start_config_rearrange, [start_config[j][i]])
        end_config_rearrange = np.append(end_config_rearrange, [end_config[j][i]])
T = np.hstack(np.subtract(end_config_rearrange, start_config_rearrange))
A = np.hstack(response_vectors)
AtA = np.dot(np.matrix.transpose(A),A)
AtT = np.dot(np.matrix.transpose(A),T)
try:
    b = np.linalg.solve(AtA,AtT)
except np.linalg.LinAlgError as err:
    if 'Singular matrix' in str(err):
        print("Need more vectors")
    else:
        raise
t_seq = np.array([])
inplane_seq = np.array([])
tilt_seq = np.array([])
num_control_signals = int((len(res_vec_control_signal)-1))
for x in range(0,num_control_signals,2):
    y_magnitude = np.abs(b[x])
    x_magnitude = np.abs(b[x+1])
    total_magnitude = np.sqrt(y_magnitude**2 + x_magnitude**2)
    alpha = np.arcsin(x_magnitude/total_magnitude)*(180/np.pi)
    if (b[x]<0):
        if(b[x+1]>0):
            alpha = 180-alpha
        else:
            alpha = 180+alpha
    elif (b[x]>0):
        if(b[x+1]<0):
            alpha = 360-alpha
    t_seq = np.append(t_seq, total_magnitude)
    inplane_seq = np.append(inplane_seq, alpha)
    tilt_seq = np.append(tilt_seq, res_vec_control_signal[x][1])
total_time = 0
for i in range(len(t_seq)):
    total_time = total_time + t_seq[i]
return start_config, tilt_seq, inplane_seq, t_seq
```